

**This Page is Inserted by IFW Indexing and Scanning  
Operations and is not part of the Official Record**

**BEST AVAILABLE IMAGES**

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ **BLACK BORDERS**
- ☐ **IMAGE CUT OFF AT TOP, BOTTOM OR SIDES**
- ☐ **FADED TEXT OR DRAWING**
- ☐ **BLURRED OR ILLEGIBLE TEXT OR DRAWING**
- ☐ **SKEWED/SLANTED IMAGES**
- ☐ **COLOR OR BLACK AND WHITE PHOTOGRAPHS**
- ☐ **GRAY SCALE DOCUMENTS**
- ☐ **LINES OR MARKS ON ORIGINAL DOCUMENT**
- ☐ **REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY**
- ☐ **OTHER:** \_\_\_\_\_

**IMAGES ARE BEST AVAILABLE COPY.**

**As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.**

## TRANSMITTAL OF APPEAL BRIEF (Large Entity)

Docket No.  
FR920000055US1

In Re Application Of: I. Agulhon

Application No.	Filing Date	Examiner	Customer No.	Group Art Unit	Confirmation No.
09/992,103	November 5, 2001	Uyen T. Le	33558	2171	7500

Invention:

Object Oriented Method And System For Transferring A File System

RECEIVED

SEP 15 2004

Technology Center 2100

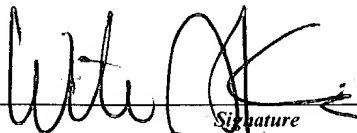
COMMISSIONER FOR PATENTS:

Transmitted herewith in triplicate is the Appeal Brief in this application, with respect to the Notice of Appeal filed on July 7, 2004

The fee for filing this Appeal Brief is: \$330.00

- ☐ A check in the amount of the fee is enclosed.
- ☒ The Director has already been authorized to charge fees in this application to a Deposit Account.
- ☒ The Director is hereby authorized to charge any fees which may be required, or credit any overpayment to Deposit Account No. 09-0463
- ☐ Payment by credit card. Form PTO-2038 is attached.

**WARNING: Information on this form may become public. Credit card information should not be included on this form. Provide credit card information and authorization on PTO-2038.**

  
Signature

William A. Kinnaman, Jr. - Attorney  
Registration No. 27,650  
IBM Corporation - MS P386  
2455 South Road  
Poughkeepsie, NY 12601

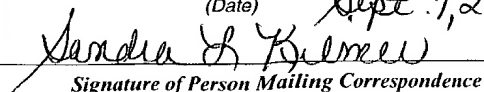
Telephone No. (845) 433-1175

Dated: September 7, 2004

I hereby certify that this correspondence is being deposited with the United States Postal Service with sufficient postage as first class mail in an envelope addressed to "Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450" [37 C.F. 1.8(a)] on

(Date)

Sept. 7, 2004

  
Signature of Person Mailing Correspondence

Sandra L. Kilmer

Typed or Printed Name of Person Mailing Correspondence

CC:

Patent

IN THE U.S. PATENT AND TRADEMARK OFFICE  
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Applicant: ISABELLE AGULHON : Group Art Unit: 2171  
Serial No.: 09/992,103 : Examiner: Uyen T. Le  
Filed: November 5, 2001 : September 7, 2004  
Confirmation No.: 7500 : William A. Kinnaman, Jr.  
Title: OBJECT-ORIENTED METHOD : International Business Machines Corporation  
AND SYSTEM FOR TRANSFERRING A : 2455 South Road, Mail Station P386  
FILE SYSTEM : Poughkeepsie, NY 12601

RECEIVED

SEP 15 2004

Technology Center 2100

APPLICANT'S APPEAL BRIEF

Commissioner for Patents  
P.O. Box 1450  
Alexandria, VA 22313-1450

RECEIVED  
2004 SEP -9 PM 4:04  
BOARD OF PATENT APPEALS  
AND INTERFERENCES

Dear Sir:

Applicant hereby submits her appeal brief in the above-identified application.

CERTIFICATE OF MAILING UNDER 37 CFR 1.8(a)

I hereby certify that this correspondence is being deposited with the United States Postal Service as first-class mail in an envelope addressed to the Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450, on September 7, 2004.

Sandra L. Kilmer  
Sandra L. Kilmer

Sept. 7, 2004  
Date:

## REAL PARTY IN INTEREST

The real party in interest is International Business Machines Corporation, the assignee of record.

## RELATED APPEALS AND INTERFERENCES

There are no related appeals or interferences.

## STATUS OF CLAIMS

Claims 1-14, constituting all claims pending in the application, stand rejected and are on appeal.

No claims have been cancelled or withdrawn.

## STATUS OF AMENDMENTS

There are no amendments after final rejection, unentered or otherwise.

## SUMMARY OF INVENTION

One aspect of the invention is directed to an object-oriented method, apparatus and program storage device for transferring a file system (Fig. 1) including folders (DIR.X, DIR.Y, DIR.Z) and data files from a source data storage controlled by a source data processing unit to a destination data storage controlled by a destination data processing unit over a transfer medium.

Referring to Fig. 4, in accordance with this aspect of the invention, at least one file object containing a data package to be transferred is built in the source data storage. Referring to Fig. 3, the file object may be a component object 10, 12, 14 (COMP) corresponding to files associated with a directory or may be an object 16 (STRUCT) corresponding to files associated with a root directory (REPOS). The file object may be built by using a file transfer tool model (Fig. 2) including a header containing the functions set directory name, get environment, set

environment, create, and install data and a body containing the items directory name, installed directory, size, version, and data package.

A descriptor file including parameters associated with the file object is created (steps 24, 36), along with an archive file including the data package (steps 28, 40). These two files are transmitted from the source data processing unit to the destination data processing unit over the transfer medium. The descriptor file and archive file may be generated by defining the descriptor file in the source data storage, setting a directory name in the descriptor file, creating the archive file from the data package, and setting remaining parameters (e.g., installed directory, size, and version) in the descriptor file.

Yet another aspect of the invention relates to an object-oriented method for receiving, in a destination storage controlled by a destination data processing unit, a file system which is transferred from a source data storage controlled by a source data processing unit. Referring to Fig. 5, in accordance with this aspect of the invention, a received descriptor file is read (steps 54, 68), and a file object defined from information contained in the descriptor file (steps 56, 70). An environment parameter of the file object is set with a value got from the descriptor file (steps 58, 72), and data associated with the file object contained in a received archive file is unarchived and installed in the destination data storage (steps 60, 74).

## ISSUES

- I. Whether claims 1-3 were properly rejected under 35 U.S.C. §§ 102(a) and (e) as being anticipated by Wiese U.S. Patent 6,108,707 (“Wiese”).
- II. Whether claims 1-14 were properly rejected under 35 U.S.C. §§ 102(a) and (e) as being anticipated by Pisello et al. U.S. Patent 5,495,607 (“Pisello”).

## GROUPING OF CLAIMS

Separately argued with respect to the rejection on Pisello et al. are (1) claims 1-6, 9 and 12; (2) claims 8, 11 and 14; and (3) claims 7, 10 and 13.

## ARGUMENT

The Examiner has rejected claims 1-3 as being anticipated by Wiese U.S. Patent 6,108,707 ("Wiese"). As an independent basis for rejection, the Examiner has also rejected claims 1-14 as being anticipated by Pisello et al. ("Pisello"). These rejections are treated separately below.

### Rejection on Wiese

Claim 1, which typifies the claims rejected on this reference, reads as follows:

1. An object-oriented method for transferring a file system including folders and data files from a source data storage controlled by a source data processing unit to a destination data storage controlled by a destination data processing unit over a transfer medium, said method comprising the steps of:
  - building in said source data storage at least one file object containing a data package to be transferred;
  - generating a descriptor file including parameters associated with said file object;
  - generating an archive file including said data package; and
  - transmitting said descriptor file and said archive file from said source data processing unit to said destination data processing unit over said transfer medium.

The other claims rejected on this reference, claims 2 and 3, depend on claim 1.

Claims 1-3 stand rejected under 35 U.S.C. §§ 102(a) and (e) as being anticipated by Wiese (paper no. 8, page 3). This rejection is untenable and should be reversed.

As is evident from its text, reproduced above, claim 1 is directed to an object-oriented method for transferring a file system including folders and data files from a source data storage controlled by a source data processing unit to a destination data storage controlled by a destination data processing unit over a transfer medium. In accordance with the invention, at least one file object containing a data package to be transferred is built in the source data storage. A descriptor file including parameters associated with the file object is generated, along with an archive file including the data package. The descriptor file and the archive file are transmitted from the source data processing unit to the destination data processing unit over the transfer medium.

Several things thus characterize applicant's invention as recited in claims 1-3. First, applicant's invention is directed to a method for transferring a file system, including folders and data files, and not just individual files of such a system as in the prior art. Second, in applicant's claimed method, both a descriptor file and an archive file are transmitted from the source data processing unit to the destination data processing unit. These two files, moreover, are interrelated in that the descriptor file includes parameters associated with a file object containing a data package to be transferred, while the archive file includes the data package itself.

Wiese describes enhanced file transfer operations in a computer system. In accordance with one aspect of the disclosure, files are transferred in the form of "chunks" rather than one file at a time (col. 4, lines 50-56). Thus, Fig. 3 shows how files 1-n are aggregated into a first chunk A with a header a and a second chunk B with a header b. However, while Wiese thus teaches the transfer of multiple files, he does not contemplate transferring information indicating the location of such files in the directory structure of a file system, and thus does not teach transferring "a file system including folders and data files" as claimed by applicant.<sup>1</sup> Further, Wiese transfers file data in the form of "chunks" rather than files that are recognizable as such by a computer system. In

---

<sup>1</sup> The Examiner asserts that applicant "makes no reference to 'location' in the specification or the claims" and that the claimed folders "merely appear in the preamble of claims 1, 9, 12 and do not seem to play any role in the body of the claims" (paper no. 8, page 2). However, while applicant may not have expressly referred to a file "location", the notion of a location in a directory tree (Fig. 1) is implicit in the concept of a file system (page 4, lines 2-9). Further, while applicant does not use the term "file system" in the body of the claims, the recitation of a "file object containing a data package" is an obvious reference back to this antecedent language and corresponds to the various components 10, 12, 14, 16 (Fig. 3) into which applicant's file system may be decomposed.

particular, Wiese does not teach transmitting an archive file and a descriptor file as claimed by applicant, where the descriptor file includes parameters associated with a file object containing a data package to be transferred, while the archive file includes the data package itself.

The Examiner argues, however, that applicant's claim limitation of generating a descriptor file including parameters associated with a file object "is met by the fact that any file includes identification data" (paper no. 8, page 3). Assuming for the sake of argument that a file has associated identification data, that does not mean that such identification data ("parameters associated with said file object") is contained in a separate file (the descriptor file) as claimed by applicant.<sup>2</sup>

The Examiner also argues that the recitation of transmitting the descriptor file and archive file over the transfer medium "is met by the fact that the archive file including identification and blocks of data is transferred to a destination location on the network" (paper no. 8, page 3). However, while Wiese may transfer archive data to a destination location, there is no suggestion of transferring separate descriptor and archive files as claimed by applicant. Again, the Examiner makes the error of double inclusion, equating each chunk transferred by Wiese both with applicant's claimed descriptor file and with applicant's claimed archive file.

For the foregoing reasons, not only are claims 1-3 not anticipated by Wiese, but they clearly distinguish patentably over that reference. Accordingly, the rejection of claims 1-3 as being anticipated by Wiese is untenable and should therefore be reversed.

---

<sup>2</sup> The Examiner argues that claims 1, 9 and 12 "do not require a separate file for the descriptor file" (paper no. 8, page 2). However, as a matter of ordinary claim construction, having already referred in claim 1 to a "file object", applicant cannot very well read the later recitation of a "descriptor file" onto the same physical element without being guilty of double inclusion. If anything, it would take extra language in the claim to include the possibility of the descriptor file being contained in the file object.

## Rejection on Pisello et al.

### 1. Claims 1-6, 9 and 12

These claims stand rejected under 35 U.S.C. §§ 102(a) and (e) as being anticipated by Pisello (paper no. 8, page 4). This rejection is likewise untenable and should be reversed.

Claims 1-3 have already been discussed above in connection with the rejection on Wiese. Claims 4-6 depend on claim 1. Claim 9 is similar to claim 1 but is directed to a system rather than a method. Claim 12 is similar to claim 1 but is directed to a program storage device rather than a method.

Pisello describes a network management system in which a virtual catalog 150.00 (Fig. 1; Table 2) provides an overview of files distributively stored across a network domain. According to the abstract, current file information is used for assisting in transferring files across a network domain.

Pisello's failings as an anticipatory reference are similar to those of Wiese, discussed above. Thus, while Pisello teaches the bulk transfer of files, he does not contemplate transferring information indicating the location of such files in the directory structure of a file system, and thus does not teach transferring "a file system including folders and data files" as claimed by applicant.<sup>3</sup> Further, Pisello does not teach transmitting both an archive file and a descriptor file as claimed by applicant, where the descriptor file includes parameters associated with a file object containing a data package to be transferred, while the archive file includes the data package itself.

The Examiner argues here, as with Wiese, that applicant's claim limitation of generating a descriptor file including parameters associated with a file object "is met by the fact that any file includes identification data" (paper no. 8, page 4). As with Wiese, however, that does not mean

---

<sup>3</sup> The Examiner's argument that applicant "makes no reference to 'location' in the specification or the claims" and that the claimed folders "merely appear in the preamble of claims 1, 9, 12 and do not seem to play any role in the body of the claims" (paper no. 8, page 2) is addressed in the discussion of Wiese above.

that such identification data is contained in a separate descriptor file as claimed by applicant.<sup>4</sup> On the contrary, in Pisello such identification data is stored, not as a separate file, but as an entry in either a local catalog 111.00, 112.00, . . . , 144.00 or a domain-wide virtual catalog 150.00 (Fig. 1; Table 2; col. 13, lines 14-55).

The Examiner also argues here, as with Wiese, that the recitation of “transmitting said descriptor file and said archive file from said source data processing unit to said destination data processing unit over said transfer medium” is met “by the fact that the archive file including identification and blocks of data is transferred to a destination location on the network” (paper no. 8, page 4). Again, this is simply incorrect, for the same reasons as before. Although Pisello may transfer an archive file to a destination location, that alone does not entail also transferring a descriptor file containing parameters associated with a file object, as claimed by applicant.

#### **Claims 8, 11 and 14**

Claim 8 as amended, which typifies this group of claims, reads as follows:

8. An object-oriented method for receiving in a destination storage controlled by a destination data processing unit a file system which is transferred from a source data storage controlled by a source data processing unit, comprising the steps of:
  - reading a received descriptor file;
  - defining a file object from information contained in said descriptor file;
  - setting an environment parameter of said file object with a value got from said descriptor file; and
  - unarchiving data associated with said file object contained in a received archive file and installing said data in said destination data storage.

---

<sup>4</sup> The Examiner’s argument that claims 1, 9 and 12 “do not require a separate file for the descriptor file” (paper no. 8, page 2) is addressed in the discussion of Wiese above.

Claim 11 as amended is similar to claim 8 as amended but is directed to a system rather than a method. Claim 14 as amended is similar to claim 8 as amended but is directed to a program storage device rather than a method.

Each of these claims is thus similar to claim 1, but is directed to the receiving side of a file transfer. More particularly, a received descriptor file containing information used to define a file object and a received archive file containing data associated with the file object are used to reconstruct the file object at the destination end of the file transfer, with an environment parameter for the file object being set with a value got from the descriptor file.<sup>5</sup> Nothing in either of the references cited even comes close to teaching this. Rather, they transfer either “chunks” of data files or archive files pure and simple, without any accompanying descriptor files. Thus, claims 8, 11 and 14 clearly distinguish over the references cited, as do dependent claims 7, 10 and 13 directed to similar subject matter.

#### **Claims 7, 10 and 13**

These claims are similar to claims 8, 11 and 14, respectively, except that they depend on claims 1, 9 and 12, respectively, rather than being independent claims. As such, they distinguish patentably over the references cited for the same reasons as claims 1, 9 and 12, as well as for the additional reasons applicable to claims 8, 11, and 14.

For the foregoing reasons, claims 1-14 are not anticipated by Pisello, but clearly distinguish patentably over that reference. Accordingly, the rejection of claims 1-14 as being anticipated by Pisello is likewise untenable and should therefore be reversed.

---

<sup>5</sup> The Examiner contends that the claimed environment parameter:

merely reads on the fact that any file has to be associated with information such as where it is stored, where it is located in the directory structure, its naming convention, the operating system that generated that file and so forth. Thus, it has to be obtained from the descriptor file in order to reconstruct the archive file in Pisello.

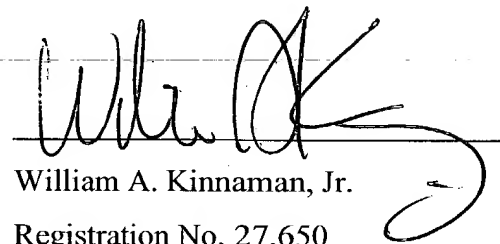
(paper no 8, page 2). This assumes, however, that Pisello transfers descriptor files separately from archive files, which is simply not the case.

### Conclusion

For the foregoing reasons, the Examiner's rejections of claims 1-14 as being anticipated by Wiese and of claims 1-14 as being anticipated by or Pisello are untenable and should be reversed.

Respectfully submitted,  
ISABELLE AGULHON

By

A handwritten signature in black ink, appearing to read 'William A. Kinnaman, Jr.', written over a horizontal line.

William A. Kinnaman, Jr.

Registration No. 27,650

Phone: (845) 433-1175

Fax: (845) 432-9601

WAK/wak

APPENDIX  
**Claims on Appeal**

1. An object-oriented method for transferring a file system including folders and data files from a source data storage controlled by a source data processing unit to a destination data storage controlled by a destination data processing unit over a transfer medium, said method comprising the steps of:
  - building in said source data storage at least one file object containing a data package to be transferred;
  - generating a descriptor file including parameters associated with said file object;
  - generating an archive file including said data package; and
  - transmitting said descriptor file and said archive file from said source data processing unit to said destination data processing unit over said transfer medium.
2. An object-oriented method according to claim 1, wherein said file object is built by using a file transfer tool model including:
  - a header containing the following functions: set directory name, get environment, set environment, create, and install data; and
  - a body containing the following items: directory name, installed directory, size, version, and data package.
3. An object-oriented method according to claim 1, wherein said file object is a component object corresponding to files which are associated with a directory.
4. An object-oriented method according to claim 1, wherein said file object is an object corresponding to files which are associated with a root directory.
5. An object-oriented method according to claim 1, wherein the steps of generating said descriptor file and generating said archive file comprise the steps of:
  - defining the descriptor file in said source data storage;
  - setting a directory name in said descriptor file;

creating said archive file from the data package; and  
setting remaining parameters in said descriptor file.

6. An object-oriented method according to claim 5, wherein said remaining parameters comprise installed directory, size, and version.
7. An object-oriented method according to claim 1, further comprising the steps of:  
reading a received descriptor file;  
defining a file object from information contained in said descriptor file;  
setting an environment parameter of said file object with a value got from said descriptor file; and  
unarchiving data associated with said file object contained in a received archive file and installing said data in said destination data storage.
8. An object-oriented method for receiving in a destination storage controlled by a destination data processing unit a file system which is transferred from a source data storage controlled by a source data processing unit, comprising the steps of:  
reading a received descriptor file;  
defining a file object from information contained in said descriptor file;  
setting an environment parameter of said file object with a value got from said descriptor file; and  
unarchiving data associated with said file object contained in a received archive file and installing said data in said destination data storage.
9. An object-oriented system for transferring a file system including folders and data files from a source data storage controlled by a source data processing unit to a destination data storage controlled by a destination data processing unit over a transfer medium, said system comprising:  
means for building in said source data storage at least one file object containing a data package to be transferred;

means for generating a descriptor file including parameters associated with said file object;  
means for generating an archive file including said data package; and  
means for transmitting said descriptor file and said archive file from said source data processing unit to said destination data processing unit over said transfer medium.

10. An object-oriented system according to claim 9, further comprising:  
means for reading a received descriptor file;  
means for defining a file object from information contained in said descriptor file;  
means for setting an environment parameter of said file object with a value got from said descriptor file; and  
means for unarchiving data associated with said file object contained in a received archive file and installing said data in said destination data storage.

11. An object-oriented system for receiving in a destination storage controlled by a destination data processing unit a file system which is transferred from a source data storage controlled by a source data processing unit, comprising:  
means for reading a received descriptor file;  
means for defining a file object from information contained in said descriptor file;  
means for setting an environment parameter of said file object with a value got from said descriptor file; and  
means for unarchiving data associated with said file object contained in a received archive file and installing said data in said destination data storage.

12. A program storage device readable by a machine, tangibly embodying a program of instructions executable by the machine to perform object-oriented method steps for transferring a file system including folders and data files from a source data storage controlled by a source data processing unit to a destination data storage controlled by a destination data processing unit over a transfer medium, said method steps comprising  
building in said source data storage at least one file object containing a data package to be transferred;

generating a descriptor file including parameters associated with said file object;  
generating an archive file including said data package; and  
transmitting said descriptor file and said archive file from said source data processing unit to said destination data processing unit over said transfer medium.

13. A program storage device to claim 12, said method steps further comprising:  
reading a received descriptor file;  
defining a file object from information contained in said descriptor file;  
setting an environment parameter of said file object with a value got from said descriptor file; and  
unarchiving data associated with said file object contained in a received archive file and installing said data in said destination data storage.

14. A program storage device readable by a machine, tangibly embodying a program of instructions executable by the machine to perform object-oriented method steps for receiving in a destination storage controlled by a destination data processing unit a file system which is transferred from a source data storage controlled by a source data processing unit, said method steps comprising  
reading a received descriptor file;  
defining a file object from information contained in said descriptor file;  
setting an environment parameter of said file object with a value got from said descriptor file; and  
unarchiving data associated with said file object contained in a received archive file and installing said data in said destination data storage.